Amendments to the Claims:

1. (Currently Amended) ~~A software tool~~ A tangible storage medium having computer readable program code that, based at least in part on a predefined association between an instruction instance in executable code and a representation of a source-level data object language construct corresponding thereto, attributes runtime events to source-level data objects describing units of data identifiable in source code.

2. (Currently Amended) ~~The software tool~~ The tangible storage medium having computer readable program code of claim 1 wherein the runtime events include sampled runtime events that statistically represent the runtime events.

3. (Currently Amended) ~~The software tool~~ The tangible storage medium having computer readable program code of claim 2 wherein the sampled runtime events include one or more of cache misses, cache references, data translation buffer misses, data translation buffer references, traps, and an event counter condition.

4. (Currently Amended) ~~The software tool~~ The tangible storage medium having computer readable program code of claim 3 wherein the event counter condition includes counter underflow or counter overflow.

5. (Currently Amended) ~~The software tool~~ The tangible storage medium having computer readable program code of claim 1 wherein the software tool includes a profiler, compiler, assembler, interpreter, or virtual machine.

6. (Currently Amended) ~~The software tool~~ The tangible storage medium having computer readable program code of claim 5 wherein the interpreter includes a byte-code interpreter.

7. (Currently Amended) ~~The software tool~~ The tangible storage medium having computer readable program code of claim 5 wherein the compiler includes one or more of an optimizing compiler and a byte code compiler.

8. (Currently Amended) ~~The software tool~~ The tangible storage medium having computer readable program code of claim 1 wherein the predefined association is included within one or more of a compiler generated code, assembler generated code, an image, and an associated separate encoding.

9. (Currently Amended) ~~The software tool~~ The tangible storage medium having computer readable program code of claim 1 wherein the instruction instance includes one or more of an instance of an instruction from a processor instruction set, an instance of an operation corresponding to a processor instruction set, an instance of a virtual machine instruction, or an instance of a byte code.

10. (Currently Amended) ~~The software tool~~ The tangible storage medium having computer readable program code of claim 1 wherein the executable code includes one or more of object code, byte code, and machine code.

11. (Currently Amended) ~~The software tool~~ The tangible storage medium having computer readable program code of claim 1 wherein the source-level data object language construct representation includes one or more of a class, a data type, a data size, a data type definition, a data structure, linked object,, and a member of a data structure, static variables, automatic variables, memory segment.

12. (Currently Amended) ~~The software tool~~ The tangible storage medium having computer readable program code of claim 1 wherein the corresponding language of the language construct includes a source-level language or an intermediate level language.

13. (Currently Amended) ~~The software tool~~ The tangible storage medium having computer readable program code of claim 1 further comprising the software tool aggregating runtime events based on the source-level data objects.

14. (Currently Amended) ~~The software tool~~ The tangible storage medium having computer readable program code of claim 13 that also displays aggregated runtime events.

15. (Currently Amended) ~~The software tool~~ The tangible storage medium having computer readable program code of claim 1 that also aggregates profile data for the code based on the source-level data object language construct representation.

16. (Currently Amended) A method of profiling code, the method comprising:

for a sampled runtime event detected in execution of the code, identifying a corresponding operation instance of the code, and

based on a predefined association between the identified operation instance and a language construct of a source-level representation corresponding to a source-level data object, attributing the detected event to the source-level data object <u>describing units of data identifiable in source code</u>.

17. (Original) The method of claim 16 wherein the predefined association is included within one or more of a compiler generated code, assembler generated code, an image, or an associated separate encoding.

18. (Original) The method of claim 16 wherein the corresponding operation instance includes one or more of an instance of an instruction from a processor instruction set, an instance of an operation corresponding to a processor instruction set, an instance of a virtual machine instruction, and an instance of a byte code.

19. (Original) The method of claim 16 wherein the code includes one or more of object code, byte code, and machine code.

20. (Original) The method of claim 16 wherein the source-level data object language construct includes one or more of a data type, a data type definition, a data structure, a data size, and a function.

21. (Original) The method of claim 16 wherein the language of the language construct includes a source-level language or an intermediate level language.

22. (Original) The method of claim 16 wherein the sampled runtime event includes one or more of a cache miss, cache reference, data translation buffer miss, data translation buffer reference, and an event counter condition.

23. (Original) The method of claim 22 wherein the counter event condition includes counter underflow or counter overflow.

24. (Original) The method of claim 16 embodied in a computer program product encoded on one or more machine-readable media.

25. (Original) The method of claim 16 further comprising aggregating profiling data for the code based at least in part on the language construct.

26. (Currently Amended) A method of profiling code, the method comprising:

associating a language construct for a source-level data object, wherein the source-level data object describes units of data identifiable in source code, with an instance of an instruction; and

based on the association between the source-level data object language construct and the instruction instance, attributing to the source-level data object language construct a sampled runtime event, which corresponds to the instruction instance, detected from execution of the code.

27. (Original) The method of claim 26 wherein the code includes one or more of object code, byte code, and machine code.

28. (Original) The method of claim 26 wherein the source-level data object language construct includes one or more of a data structure, classes, variable instances, objects, a member of a data structure, and a statically linked object.

29. (Original) The method of claim 26 wherein the instruction instance includes a load type instruction.

30. (Original) The method of claim 26 wherein the language construct includes one or more lexical tokens.

31. (Original) The method of claim 30 wherein the lexical tokens include one or more of identifiers and literals.

32. (Original) The method of claim 26 wherein the source-level data object is represented in a source-level language or an intermediate level language.

33. The method of claim 26 wherein the sampled runtime event includes one or more of a cache miss, cache reference, a data translation buffer miss, data translation buffer reference, and a counter condition event.

34. (Original) The method of claim 33 wherein cache miss includes a data cache miss, an instruction cache miss, a unified cache miss, and an external cache miss.

35. (Original) The method of claim 33 wherein cache references includes one or more of a data cache reference, an instruction cache reference, a unified cache reference, and an external cache reference.

36. (Original) The method of claim 33 wherein the counter condition event includes a counter overflow event or a counter underflow event.

37. (Original) The method of claim 26 further comprising backtracking from a second instruction instance to the instruction instance after detecting the sampled runtime event.

38. (Original) The method of claim 26 embodied in a computer program product encoded on one or more machine-readable media.

39. (Original) The method of claim 26 further comprising aggregating profile data for the code based at least in part on the language construct.

40. (Currently Amended) A computer program product for profiling code, encoded in one or more computer readable media, the computer program product, when executed, performs operations comprising:

identifying an instruction instance that corresponds to a runtime event; and

attributing the runtime event to a language construct representation of a source-level data object, the language construct representation having been associated with the instruction instance, wherein the source-level data object describes units of data identifiable in source code.

41. (Original) The computer program product of claim 40 wherein the instruction instance includes one or more of a microinstruction and a macroinstruction.

42. (Original) The computer program product of claim 40 wherein the runtime event includes a sampled runtime event that statistically represents the runtime event.

43. (Original) The computer program product of claim 42 wherein the sampled runtime event includes one or more of a cache miss event, a cache reference event, a data translation buffer reference event, a data translation buffer miss event, and a counter condition event.

44. (Original) The computer program product of claim 40 wherein the source-level data object language construct representation includes a data type, a type definition, a data structure, and a member of a data structure.

45. (Original) The computer program product of claim 40, when executed, further performs operations comprising aggregating profile data based on the language construct representation.

46. (Currently Amended) An apparatus comprising:

a computer readable encoding corresponding to an instruction sequence; and

means for correlating a sampled runtime event and a language construct that corresponds to a source-level data object, wherein the source-level data object describes units of data identifiable in source code, that has an association with an instance of an instruction of the instruction sequence, the instruction instance corresponding to the sampled runtime event.

47. (Original) The apparatus of claim 46 wherein the computer readable encoding encodes a source-level representation corresponding to the instruction sequence.

48. (Original) The apparatus of claim 46 wherein the computer readable encoding encodes the instruction sequence.

49. (Currently Amended) An apparatus comprising:

a set of one or more processors;

a memory coupled with the set of processors; and

a machine-readable media coupled with the set of processors, the machine-readable media having stored therein a set of data profiling instructions to cause the set of processors to, attribute a runtime event to a language construct that corresponds to a source-level data object, wherein the source-level data object describes units of data identifiable in source code, that is associated with an instance of an instruction, which corresponds to the runtime event.

50. (Original) The apparatus of claim 49 wherein at least one of the set of processors includes a data cache.

51. (Original) The apparatus of claim 49 wherein the machine-readable media includes propagated signal, optical storage, or magnetic storage.

52. (Original) The apparatus of claim 49 wherein the set of data profiling instructions further cause, when executed, the set of processors to aggregate profile data for the code based on the source-level data object language construct.